Table 5-3 Table Format Query Options

Command	Description
Polling	Polling sets the number of seconds between updates when a query is performed. For example, if polling is set to 5, once the query is performed, the data will be updated every 5 seconds. Polling can be set from 0 to 6000 seconds. A zero (0) means polling takes place only once when the Perform or Start button is pressed.
Community	Community is the name used by an SNMP device to restrict access to MIB variables. Entering a valid community name gives you read-only access to MIB variables.
Set Community	Set Community is the name used by an SNMP device to restrict access to device MIB variables. Entering a valid Set Community name gives you read and write access to device MIB variables.

*To permanently set community values, choose **Customize Device Access** from the **Options** menu.*

Performing a Query

After you enter the appropriate information for the device, variable, and options, click on **Perform** to execute the query. If a polling value other than zero was set, the table or graph will be updated periodically.

Example 1: Displaying an SNMP Table Variable

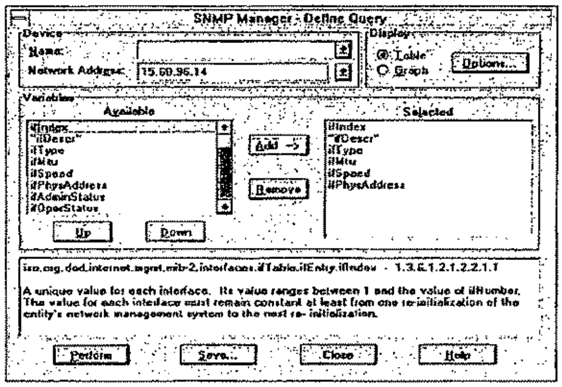
The following screen shows a sample query with results to be displayed as a table. The {ifTable} variable is selected. Since the selected variable is an SNMP table, it is the only variable selected.

When you select the {ifTable} variable, you have two options:

- You can display the {ifTable} variable with a column for every variable in the table.
- You can display a subset of the {ifTable} variables. To do this, click on {ifTable} in the Available box and then click on the **Down** button. [ifentry] appears, click on the **Down** button again. A list of column names appears. To get a description, click on a column name. To add one or more column names to the Selected box, click on one column name or Ctrl-click to select multiple column names,

and then click on **Add**. When you perform a query, only the selected columns appear in the table.

For this example, polling was set to zero so that the polling only occurs once. If polling were set to 10 seconds, the values in the table would be updated from the device every 10 seconds while this window is displayed on the screen.



Name:
Network Address: 15.60.96.14

● Table
○ Graph

Available
- ifIndex
- "ifDescr"
- ifType
- ifMtu
- ifSpeed
- ifPhysAddress
- ifAdminStatus
- ifOperStatus

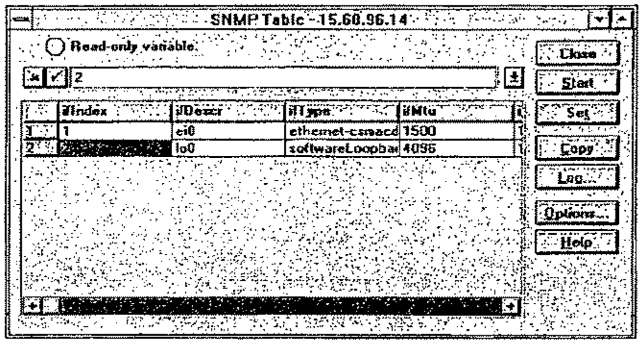
Selected
- ifIndex
- "ifDescr"
- ifType
- ifMtu
- ifSpeed
- ifPhysAddress

iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifIndex - 1.3.6.1.2.1.2.2.1.1

A unique value for each interface. Its value ranges between 1 and the value of ifNumber. The value for each interface must remain constant at least from one re-initialization of the entity's network management system to the next re- initialization.

CH6-20

Performing this query results in the following table:



○ Read-only variable.

2

	ifIndex	ifDescr	ifType	ifMtu
1	1	ei0	ethernet-csmacd	1500
2	2	lo0	softwareLoopbac	4096

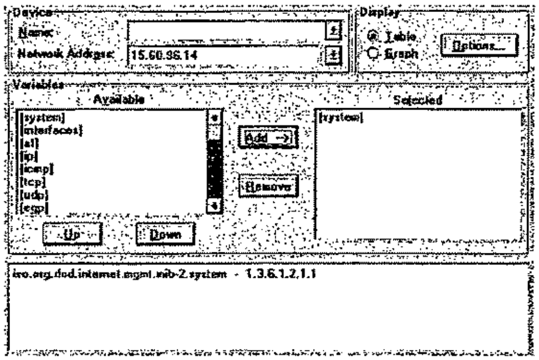
CH6-21

Table 5-4 Table Display Options

Button	Description
Start	Polls the device. The button will change to STOP until the poll is completed or until pressed again (which cancels the polling). If the polling time is set to anything other than zero, the polling will continue until the STOP button is pressed.
Set	Allows a variable's value to be changed if the device permits the value to be changed.
Copy	Copies selected cells from the table to the clipboard in tab delimited format. You can paste this information into other applications, such as word processors, spreadsheets, and databases. The variable names corresponding to the selected cells are also included on the clipboard.
Log	Allows the current data to be saved to a named file in comma delimited format, replacing the contents of an existing file. Each time you log new data obtained by polling, the log file is appended. The log file includes the name and address of the logged device followed by a blank line, the names of the variables logged, and a time stamp.
Options	Allows you to set the polling rate and community.

Example 2: Displaying Non SNMP Table Variables

The following screen also shows a sample query with results to be displayed as a table. The variables selected do *not* include an SNMP table, so multiple variables can be selected.



Name:

Network Address: 15.60.36.14

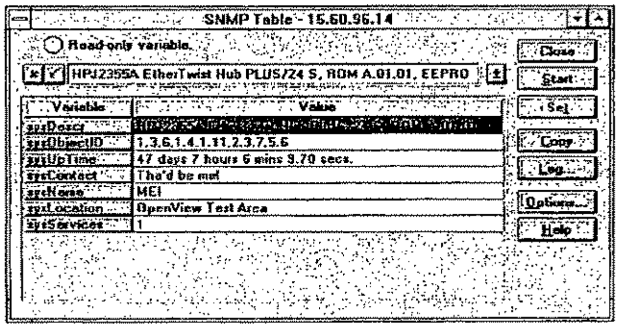
● Table
○ Graph

Available
[system]
[interfaces]
[at]
[ip]
[icmp]
[tcp]
[udp]
[egp]

Selected
[system]

iso.org.dod.internet.mgmt.mib-2.system - 1.3.6.1.2.1.1

Multi-variable results displayed in table form:



SNMP Table - 15.60.96.14

○ Read-only variable.

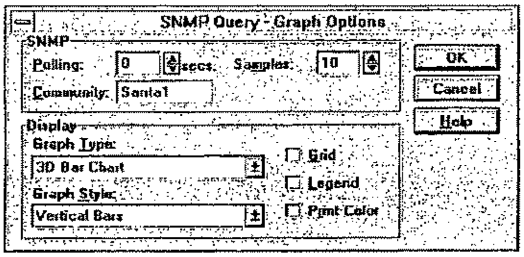
HPJ2355A EtherTwist Hub PLUS/24 S, ROM A.01.01, EEPRO

Variable	Value
sysDescr	[illegible]
sysObjectID	1.3.6.1.4.1.11.2.3.7.5.6
sysUpTime	47 days 7 hours 6 mins 9.70 secs.
sysContact	Tha'd be me!
sysName	MEI
sysLocation	OpenView Test Area
sysServices	1

CH6-24

Displaying a Query as a Graph

If single value variables are selected, you can view your query results as a graph. After selecting to display your query results as a graph, you can select several options with the Options button in the Display box.



SNMP Query - Graph Options

SNMP

Polling: 0 secs. Samples: 10

Community: Santa1

Display

Graph Type: 3D Bar Chart

Graph Style: Vertical Bars

□ Grid

□ Legend

□ Print Color

CH6-25

Table 5-5 Graph Display Options

Command	Description
Polling	Polling sets the number of seconds between updates when a query is performed. For example, if polling is set to 5, once the query is performed, the data will be updated every 5 seconds. Polling can be set from 0 to 6000 seconds. A zero (0) means polling takes place only once when the Perform or Start button is pressed. A non-zero value will plot the variables against time.
Samples	You may set a sample size when displaying the query as a graph and polling is set to one or greater. For example, if the sample size is set to 10 (the default), the graph will display the last ten sample points – one sample point for each poll.

Community	Community is the name used by an SNMP device to restrict access to MIB variables. It is often used as a password and is obtained from your network manager.
Graph Type	Select from several different graph types.
Graph Style	Select from a variety of presentation styles based on the Graph Type selected.
Grid	You can add a grid to the graph. No grid is the default.
Legend	You can add a legend to the graph. Having a legend is the default.
Print Color	If you have a color printer, you can print the graph in color. No color is the default (output is shades of gray). NOTE: The Print option is available after the Perform button is pressed.

Performing the Query

As with a table, enter the information into the Define Query window, then click on **Perform** to execute the query. In this case, a graph will be displayed on the screen. If you set a polling value other than zero, the graph will be updated periodically.

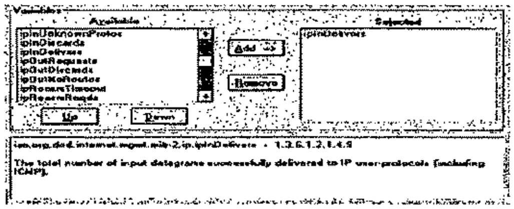
Table 5-6 Graph Display Options

Function	Description
Start	Polls the device. The button will change to STOP until the poll is completed or until pressed again (which cancels the polling). If the polling time is set to anything other than zero, the polling will continue at its designated interval until the STOP button is pressed.
Print	Prints the graph. If you have a color printer, select Print Color from the options button.
Copy	Copies the entire graph as a bitmap to the clipboard. You can paste it into drawing programs or word processors.
Log	Allows the current data to be saved to a named file in comma delimited format, replacing the contents of an existing file. Each time you log new data obtained by polling, the log file is appended. The log file includes the name and address of the logged device followed by a blank line, the names of the variables logged, and a time stamp.

Example 3: Displaying a Query as a Graph

Display variable 1.3.6.1.2.1.4.9 (ipINDelivers). By selecting the Options button in the Display box, a polling rate and a sample size can be set.

In this example, we set polling to 10 seconds and the sample size to 5.

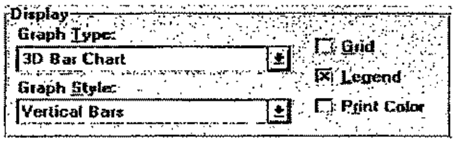


iso.org.dod.internet.mgmt.mib-2.ip.ipInDelivers - 1.3.6.1.2.1.4.9

The total number of input datagrams successfully delivered to IP user-protocols (including ICMP).

CH6-28

The type of graph is set to 3D Bar Chart



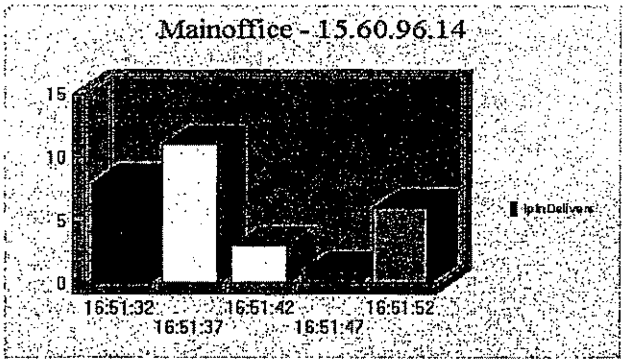
Graph Type: 3D Bar Chart

Graph Style: Vertical Bars

☐ Grid
☒ Legend
☐ Print Color

CH6-23

Click on **Perform** to generate the graph.



CH6-27

The x axis is time, since the polling is set to 10 (1 or greater). If you set the polling to zero, the x axis would list the variables selected in the Define Query window. The y axis gives the values of the variables. If you wait a minute before you STOP the polling, you can see how polling and sample size work. Because polling was set to 10 seconds, every ten seconds after the perform button was pressed, the device was polled. The sample size was set to 5, so the last five samples are displayed.

Changing a Variable's Value

You can change Read/Write variables when they are displayed as a table.

To change a variable's value, follow these steps:

1. Select the value you want to modify from the Variable column.

   The value will appear in the edit field above the table. If the value cannot be set, the status line displays the message: **Read-only variable**.

2. Make the change you want to the value in the edit field.

   To cancel the changes, press the Cancel button.

3. Press the Update button.

   The changed value will be displayed in the table. The value to be changed will be displayed in red. The change has not yet been made to the device.

4. Repeat steps 1 through 3 for each value you wish to modify.

5. To update the device with the changes, press the Set button.

   If you click on the **Start** button *before* the Set button, all changes to the table values are canceled without changing the device's values. Any table values that were modified will be updated from the device (by polling).

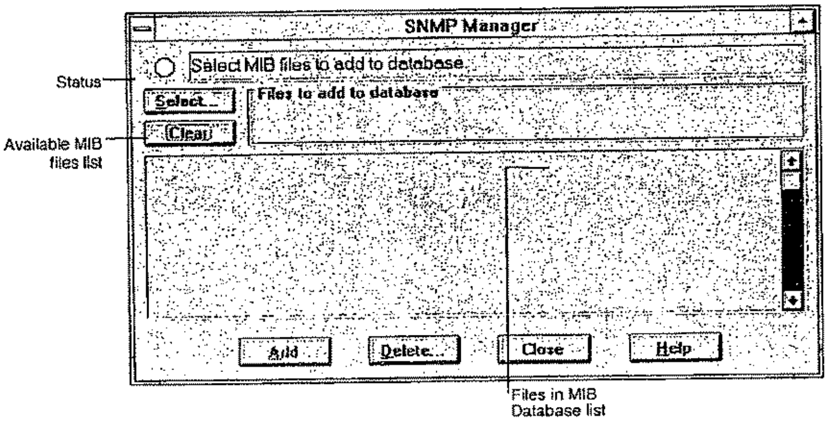
Managing the SNMP Manager Database

The Manage Database command is accessed from the SNMP Manager in the OpenView Control menu. Manage Database accesses a compiler that adds MIBs to the Manager's database. MIBs can contain additional information which change the list of variables displayed in the Define Query window. To use the compiler to modify the existing OpenView MIB database, an SNMP v.1 or v.2 definition file is required. The definition files will normally have the file extension .MIB. After the definition file has been compiled into the OpenView database, the new variables will appear in the variable list in the Define Query window. The MIB compiler can extract device trap definitions to be used with the Trap Manager.

The manage database window consists of several parts:

- The status line
- The Available MIB files list

- The Files in MIB Database list



Status Line The manage database window includes a status line. The status indicator and a line of text indicate the current status. The indicator displays one of three colors:

Table 5-7 Indicator Color Legend

Color	Description
Green	indicates a ready state and, where appropriate, a successful compile
Yellow	indicates the compile is in progress
Red	Indicates an error state

Available MIB Files This section of the Manage Database window allows you to select the file(s) to be compiled.

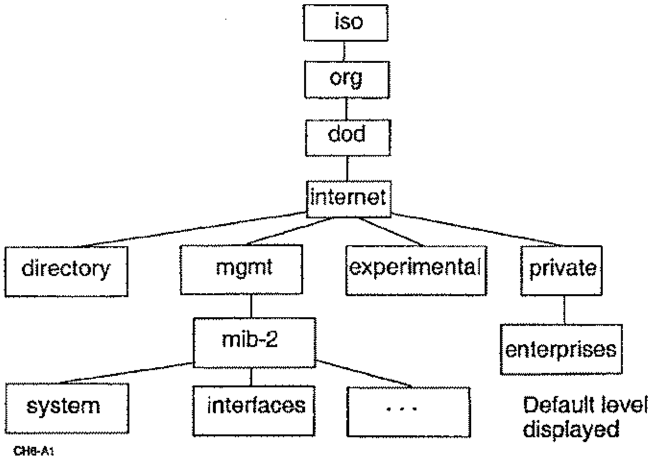
- The **Import** button allows you to browse the directory for the desired MIB files.
- The **Available MIB Files** box lists all selected files to be compiled. The MIB files may be SNMP v.1 or v.2. If a definition file is a dependent (a child) of another definition file (parent), the parent MIB files must be previously compiled or currently selected. Missing dependencies will generate a prompt for the MIB file.
- The **Add** button compiles the selected files in the Available MIB Files list.

Files in MIB Database After MIB files are successfully compiled, they will be listed in this section of the manage Database window. This list of files is maintained in a file in .INI format.

- The Clear Database button allows you to delete all the files in the MIB database.

MIB Structure

One of the keys to using the SNMP Manager is understanding the structure of the MIBs. This understanding is important when navigating through the variable tree. It is also important when compiling your own MIBs – some MIBs have dependencies on MIBs higher in the tree. The MIB structure of OpenView is organized as shown in the map below.



There are two areas where you might commonly add MIBs:

- MIBs dependent upon MIB-2
- Vendor-specific private MIBs

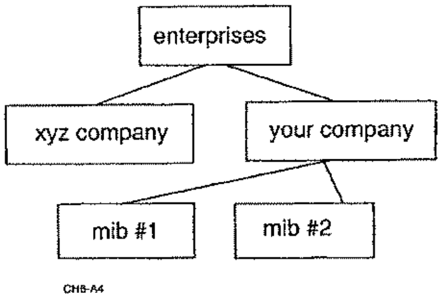
MIBs Dependent Upon MIB-2 MIB-2 dependent MIBs, such as **rmon**, would be added to the structure under the MIB-2 group:

In this example, the MIB defining **rmon** would have to be compiled. However, its dependent relationship to MIB-2 requires that MIB-2 *also* be compiled

To accomplish this, use **Manage Database** to **Add** both MIBs. Select the MIB for MIB-2 first and then the MIB for **rmon**. OpenView provides the MIBs for both MIB-2 (RFC1213.MIB) and **rmon** (RFC1271.MIB). These files are in the MIBS directory under OpenView (ex. C:\OV\MIBS).

Vendor-Specific Private MIBs

To add a private MIB, add it under enterprises as illustrated below.



The structure illustrated above is a basic one. You could create an entire tree hierarchy under the group you create for your company.

Selecting MIB Files to Add to the Database

OpenView comes with a MIB-2 SNMP database compiled from RFC1213.MIB. This file is located in the MIBS directory of OpenView.

To add information from other MIB files to this database follow these steps:

1 Select the file(s) you want compiled in the Available MIB Files box of the Manage Database window.

2 Click the Add button.

The compile may take several minutes to complete. Once started, you cannot terminate the process.

Adding Device Information

Once you have selected the MIB files to be compiled, press the **Add** button. **Add** appends the MIBs to the existing OpenView database. **Add** overwrites duplicate entries.

To completely rebuild the OpenView MIB variables list, click on **Clear Database** to delete the currently compiled MIBs. You can then select the appropriate MIB files and compile a new database.

Clearing SNMP Database

The **Clear Database** button deletes the *entire* OpenView SNMP database and its associated list of variables in the Define Query window.

6 In Case Of Difficulty

HP OpenView for Windows is designed to run several different OpenView applications simultaneously under Microsoft Windows. There are many possible settings and configurations for the WIN.INI and OVWIN.INI files, and a variety of applications and devices that can be used. Be sure to refer to the *Installation Tips* and *Troubleshooting Guide* files in the OpenView program group. These files contain the latest configuration and troubleshooting information.

Things You Should Know

Diagnosing problems requires that you provide as much configuration information as possible to software support personnel. To aid in diagnosing and correcting problems you should be able to provide the following:

- Version of HP OpenView, MS Windows, and MS DOS
- Network software you are using.
- List of OpenView applications installed
- Copies of your `ovwin.ini`, `win.ini`, `system.ini`, `config.sys`, and `autoexec.bat` files
- Error message displayed
- Brief description of what you were doing at the time of the error
- Map files (.ovi, .ovd, .ovm, and .adl)
- Log files (defined in the **Log=** entry in **ovwin.ini**)
- Types of nodes in your map
- Device type and vendor of a particular device that is causing the problem
- `ov\ovfiles\devices, cdbnodes, cdbnodep`

You may not be able to provide all this information, but the more you can provide, the more quickly your software support representative can solve the problem.

What You Should Do

If you think that the problem occurred with a particular OpenView application, refer to the application documentation for information on error messages and support procedures. Additional troubleshooting information is given in readme and other text files available in the OpenView Program Group.

If the problem is with OpenView or with functions described in this manual, refer to the error message descriptions in on-line help. If you are not able to run OpenView to access help, access help directly using the help icon in the OpenView program group.

Technical support is available for the Workgroup Node Manager at the HP Customer Response Centers. Check the HP OpenView for Windows Web site for the latest support information www.hp.com/go/ovwin.

Phone support: HP Customer Response Center

	Phone	Fax
U. S. & Canada	(970) 635-1000	
Europe		
English	(+31 20) 581-3330	
Dutch	(+31 20) 581-3331	
French	(+31 20) 581-3332	
German	(+31 20) 581-3333	
Asia / Pacific		
Australia	+61-3-9272 2895	+61-3-9898 7831
Korea	+82-2-769 0114	+82-2-3270 7084
Taiwan	+886-2-717-9651	+886-2-715-3107
China	+86-10 65053888 ext 5901	+86-10 65924566
Faxback - HP First		
U.S.	(800) 333-1917	
non U.S.	(208) 334-4809	

Other Countries, contact your local HP office or dealer.

OVMDump Diagnostic Tool

OVMDump is a utility tool in the default directory C:\OV\DIAG. OVMDump allows you to look at information in an .OVM file. Simply execute OVMDump from this directory, specifying the target .OVM file, and a file to contain the output produced from OVMDump. For example,

```
C:\OV\DIAG\OVMDUMP T1MAP.OVM > T1DUMP
```

Your output will have the following format:

```
            Header
 File Name:                                 connect.ovm
Date of modification:                 Mon Sep 04 14:28:52 1992
File Version                                7
Number of Symbols                           15
Number of Names:                      6
Number of Labels:                           5
Number of Submaps:                          1
Current submap context:               0

OLD SYMBOL SUMMARY STATISTICS

SYMBOL ID                       Description                Count
--------------------------------
0x1400                          Submap                       1
0x1209                          Line 8                       9
0x1327                          Large Computer 1             1
0x1329                          Medium Computer 2            1
0x1337                          T1 Mux                       1
```

Depending on the error, this information may be valuable to your software support representative.

OpenView Backup

This section describes how to use the OpenView Data-File Backup utility to backup data files such as configurations and maps for archival purposes.

The HP OpenView Backup utility allows the user to backup, archive, and restore OpenView data files.

To backup or restore OpenView data files:

1 Close OpenView. (Backups cannot be performed while OpenView files are open.)

2 From the Start menu, select Programs.

3 Select HP OpenView and then HP OpenView Data-File Backup.

In the Backup Options box, selecting changed data files will backup only files that have changed since the last OpenView Backup. Selecting All data files will do a full backup of HP OpenView data files with a file date newer than OVWIN.Exe.

In the Restore Options box, selecting Overwrite files will restore all files in the archive. Selecting Overwrite older files will restore only data files that are more recent than those found in the OpenView directory structure.

4 If desired, change the backup filename in the Archive Name field.

5 Select the desired Backup or Restore Options (described above).

6 Click the Backup or Restore button, as appropriate.

7 Click Done to exit.

The OpenView Backup File menu lists options to view, rename, and delete existing archives.

The Archive View option lists the files contained within the archive, along with the total number of files in the archive. Selecting one of the filenames in the scroll box displays information about that file in the File Information box. Note that File Information does not provide information about the archive file itself.

The Archive Rename option allows you to rename an archive file.

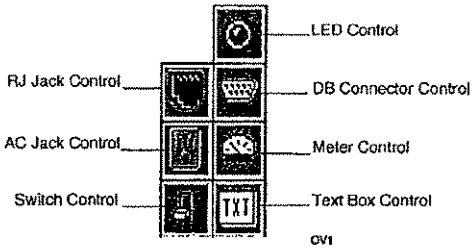
The Archive Delete option allows you to delete an unwanted archive file.

7

Custom Controls

Included in the Workgroup Node Manager is a set of Visual Basic custom controls designed for OpenView. These controls allow you to create Visual Basic applications to help you manage your network. The controls allow you to access and monitor SNMP devices. All device Mib-variables are read only. The following description assumes that you are familiar with Visual Basic and network operation using SNMP and MIB variables. The remainder of this chapter provides a description of the custom controls and program examples.

The OpenView SNMP custom controls provide visual indications of the values of SNMP variables for any SNMP device. The custom control set consists of the following elements:



Each control can be configured to poll an SNMP variable on a device and display the variable graphically. You can also display the value in a box beneath the control.

The controls also have an Alarm capability which allows you to set low and high thresholds which will cause the control to change from normal to alarm colors when those thresholds are exceeded.

With the exception of the Trap property and menu commands, these controls ***do not*** *interact with the OpenView map.*

The XOV custom control is used to terminate your program when OpenView terminates. This control is described later in this chapter.

Code Components

The code components include:

- HPOVCC.VBX – OpenView SNMP custom controls.
- XOV.VBX – OpenView custom control.
- XOVSRVR.EXE – OpenView application to interface OpenView to the custom controls.

HPOVCC.VBX – This file contains the OpenView SNMP custom controls.

XOV.VBX – The XOV Visual Basic custom control initiates and terminates communication with XOVSRVR. This may be used to shut down your application when OpenView terminates. Only one instance of this Control can be used in your program.

XOVSRVR.EXE – XOVSRVR is an OpenView application. This program is started by OpenView and communicates with your Visual Basic application.

During its initialization, XOVSRVR sets the menus and registers symbols for the Visual Basic applications. These are defined by entries in the XOVSRVR.INI file. It can also start the Visual Basic applications. The applications can be started when OpenView starts or they can be started from an OpenView menu command.

Control Description

The following is a list of the unique properties of the OpenView SNMP custom controls.

Control	Description	Properties	
AC Connector	The AC Connector indicates the power line status. It displays a standard Three-Blade power cord receptacle.	Orientation: Integer This property selects the orientation of the AC connector:	▪ Up (0) ▪ Down (1) ▪ Right (2) ▪ Left (3)
DB Connectors	The DB-x represents communications status through a DB-x connector.	Orientation: Integer This property selects the orientation of the DB-x connector:	▪ Up (0) ▪ Down (1) ▪ Right (2) ▪ Left (3)
		DBConnector: Integer This property selects the type of DB-x connector displayed:	▪ DB-9 (0) ▪ DB-15 (1) ▪ DB-25 (2)
LED	The LED displays SNMP values through a simple binary "LED".	DisplayType: Integer Sets the type of LED to display:	▪ Round (0) ▪ Square (1)

Continued

Meter	The Meter provides a method to display integer and counter variables through a meter type display.	Orientation: Integer This property selects the orientation of the meter:	■ Up (0) ■ Down (1) ■ Right (2) ■ Left (3)
		Max: Integer This property sets the highest value that will be displayed on the gauge. It sets the upper limit of the display.	
		Min: Integer This property sets the lowest value that will be displayed on the gauge. It sets the lower limit of the display.	
		ScaleType: Integer This property selects the type of scale used for the meter:	■ Linear (1) ■ Logarithmic (2)
		GaugeType: Integer This property selects the type of meter that will be displayed:	■ Bar (0) ■ Needle (1) ■ Dial (2)
RJ-45	The RJ-45 displays a RJ-45 Jack in various orientations and with color feedback of status.	Orientation: Integer This property selects the orientation of the RJ-45 connector:	■ Up (0) ■ Down (1) ■ Right (2) ■ Left (3) ■

Switch	The Switch displays a variety of switches with binary attributes of an SNMP device.	Orientation: Integer This property selects the orientation of the switch. The orientations, with respect to "on" are:	▪ Up (0) ▪ Down (1) ▪ Right (2) ▪ Left (3)
		SwitchType: Integer This property selects the type of switch that will be displayed. The type is:	▪ Slide (0)
Text Box	The Text Box displays variable values. The Text Box represents text versions of numeric values and ASCII or hexadecimal versions of Octet Strings.		

Properties

All properties except those noted are available at both design time and run time. In addition to the standard Visual Basic properties the OpenView SNMP custom controls have the following properties:

- About
- Alarm
- AlarmColor
- AlarmValues
- Community
- DataType
- Enabled
- ErrorColor
- ForeColor
- InstanceIndex
- MaxReset
- MaxThreshold
- MibVariable
- MinReset
- MinThreshold
- NetAddr
- NormalColor
- NormalValues
- PollRate
- Retries
- ShowValue
- StringValue
- TableIndex
- TimeOut
- Trap
- Value
- VariableType

Summary of Control Properties

The following table lists the properties supported by the OpenView custom controls.

Name	Type	Description
(About)	String	The About property is only available at design time. Clicking on it in the control's property window brings up the standard About box.
Alarm	Enum	Specifies whether the control should check any of the threshold properties for alarm conditions, and if so, which thresholds to evaluate. Defaults to Disabled. (Disabled (0), Min/Max Thresholds (1), Norm/Alarm Values (2).

AlarmColor	Long	Sets the color used to draw the control if the value is equal to any of the listed AlarmValues or thresholds. Defaults to Red.
AlarmValues	String	The AlarmValues property contains a comma-delimited list of values consisting of specific states of the variable that represent alarm conditions (i.e. 5, 6, 7, 200, 201, 1000, 1001). Used when Alarm=Norm/AlarmValues
BackColor		The standard VB BackColor property.
BorderStyle		The standard VB BorderStyle property.
Community	String	The Community property contains the name of the community to be used in a Get request. It may be set at design time or runtime. The default community name is "public".
DataType	Integer	This property will allow the user to specify how the variable data should be used. Either the Absolute or the Delta (difference between polls) value can be selected. Delta is normally used for counter variables. The DataTypes are: • Absolute (0) • Delta (1)
Enabled	Integer	Standard property to enable control. If false, no SNMP request will be made. Setting this value to TRUE will cause an initial request to be made, and will start polling if PollRate is greater than zero. Enabled is set to FALSE when an error occurs, and must be explicitly set to TRUE to restart polling. This property should be set to TRUE after changing NetAddr or Community to start polling.
ErrorColor	Long	Color used to draw the control when an error occurs. Defaults to Dark Gray.
ForeColor	Long	Color used to draw the foreground of the control. Defaults to Black.
Height		The standard VB Height property.
Index		The standard VB Index property.
InstanceIndex	Integer	Specifies the index of the MibVariable instance that is to be polled. Defaults to zero (0). Not normally changed by the user, this is automatically set by the control at runtime when TableIndex is used.
Left		The standard VB Left property.
MaxReset	Long	Defines the value that the variable must reach, after crossing the threshold, to reset the alarm condition. If the variable drops below this value, and there is an outstanding alarm condition, the control will be displayed in NormalColor if Alarm=Max/Min Thresholds.
MaxThreshold	Long	Defines the upper limit for the variable. If the variable exceeds this value, and there is not an outstanding alarm condition, an alarm event will be generated and the control will be displayed in AlarmColor if Alarm=Max/Min Thresholds. If the Trap property is set to TRUE an OpenView Alarm will be sent to the AlarmManager.

MibVariable	String	MIB object name to be retrieved from device specified in NetAddress. This name should be of the standard MIB format. (e.g. 1.3.6.1.2) These names may be found in the MIB specification, or examined using the Define Query dialog in OpenView's SNMP Manager.
MinReset	Long	Defines the value that the variable must reach, after crossing the threshold, to reset the alarm condition. If the variable rises above this value, and there is an outstanding alarm condition, the control will be displayed in NormalColor if Alarm=Max/Min Thresholds. If the Trap property is set to TRUE the corresponding OpenView Alarm will be cleared from the AlarmManager.
MinThreshold	Long	Defines the lower limit for the variable. If the variable drops below this value, and there is not an outstanding alarm condition, an alarm event will be generated and the control will be displayed in AlarmColor if Alarm=Max/Min Thresholds. If the Trap property is set to TRUE an OpenView Alarm will be sent to the AlarmManager.
MousePointer		The standard VB MousePointer property.
Name		The standard VB Name property.
NetAddress	String	Network address for SNMP Get. This should be in standard IP or IPX format (e.g. 1.2.3.4)
NormalColor	Long	Sets the color used to draw the control if the value is equal to any of the listed NormalValues or thresholds. Defaults to Green.
NormalValues	String	The NormalValues property contains a comma-delimited list of values consisting of specific states of the variable that represent normal conditions (i.e. 5, 6, 7, 200, 201, 1000, 1001). Used when Alarm=Norm/Alarm Values.
PollRate	Integer	Rate in seconds to poll the specified device for the specified variable. If zero then no polling will take place - the device will get a single request for the variable when the control is enabled.
Retries	Integer	Retries is used to set the number of SNMP request retries that should be attempted by the control before failing with an error event. -1 is the default which will use the system default.
ShowValue	Boolean	If TRUE then a text box is displayed at the bottom of the control to show the current StringValue.
StringValue	String	A string version of the variable's current value, which may be displayed at the bottom of the control if ShowValue is TRUE. If the variable is an integer or counter then this contains a string representation of Value; if the variable is a string, object name, or network address then this will contain a string version of the variable and Value will be zero.
TableIndex	Integer	When set to a non-zero value, in the next poll cycle the control will perform this many GET-NEXT requests on the MIBVariable. If no errors are encountered, the control will set the InstanceIndex to the returned table index value.
Tag		The standard VB Tag property.
Timeout	Long	The Timeout property determines how long in milliseconds the SNMP library will wait for a response before timing out. It may be set at design time or runtime. The default value is -1 which will cause the library to use the default system timeout value.

Top		The standard VB Top property.
Trap	Boolean	Trap is used to tell the control whether to send an SNMP Trap packet when ever alarm conditions are set or cleared. If set to False (default) no SNMP Traps will be sent. If set to True, SNMP Traps will be sent to OpenView whenever a threshold is crossed that creates an alarm condition, and also whenever a reset value is crossed causing an alarm condition to be cleared. These traps will result in alarms in the OpenView Alarm Log.
Value		The standard VB Value property.
VariableType	Enum	Identifies the type of SNMP variable that was polled. This property will contain the variable type of the returned SNMP variable. It will contain the ASN type number of the variable. 1 - Bool 2 - Integer 3 - BitStr 4 - OctetStr 5 - Null 6 - ObjectID
Visible		The standard VB Value property.
Width		The standard VB Value property.

Events These events are designed to allow you to implement your own event handling routines in your program. All of the OpenView SNMP custom controls support the following Visual Basic events:

Event	Description	Visual Basic
Alarm	*controlName*_Alarm is generated anytime the MIB variable being polled changes to cross any threshold value or match any of the values in AlarmValues depending on the setting of Alarm. The single parameter passed gives the current value of the variable. You can use this event to write other alarm handling procedures.	Sub *controlName*_Alarm(Value As Long)
Click	*controlName*_Click is the same as the standard Visual Basic Click event	
Error	*controlName*_Error is generated anytime an error is encountered by the control. This could include SNMP errors such as TimeOut, NoSuchName, WrongCommunity, etc. The single parameter passed gives the value of the error encountered.	Sub *controlName*_Error(ErrorNum As Integer)

Update	*controlName*_Update is generated every time a successful poll cycle completes. The single parameter passed gives the current value of the variable.	Sub *controlName*_Update(Value As Long)

Basic Operation In order for any of the OpenView SNMP custom controls to work you must at a minimum set the following properties:

- NetAddr
- MibVariable
- PollRate
- Community

These properties are defined as follows:

NetAddr – This must be set to a valid Network Address. This address will be used in all SNMP requests.

MIBVariable – This must be set to the OID of the MIB Variable to be polled. This property will not include the ".0" to indicate a leaf node, or the index if the variable represents a table entry.

PollRate – This will be set to indicate the number of seconds between polls.

Community – This will be set to the string value of the community string for accessing the device at NetAddr.

A simple application could have any one of the controls on a Visual Basic form with these four properties set and it would begin to display SNMP data.

Example: Place an SNMP Text box control on a form and then assign the following values:

NetAddr:	**192.63.36.1**	(Or the device of your choice)
MibVariable:	**1.3.6.1.2.1.1.1**	(sysDescr in MIB II)
PollRate:	**10**	(Every 10 seconds)
Community:	**public**	(Or the SNMP Community name for the device)

If you run the application, the control will poll the device at NetAddr every PollRate seconds and retrieve the device's System Description in the MibVariable (sysDesc from MIB II).

Data Properties

Properties that make it easier to work with SNMP data values are as follows:

- DataType
- ShowValue
- Value
- StringValue
- VariableType

DataType – can be set to Absolute or Delta. This tells the control whether to display the actual value that was returned from the SNMP device
(Absolute) or the difference in the variable since the last poll (Delta).

The Text Box control could be used to poll the UDPInDatagrams of a device. If you set the DataType of the control to Absolute, you will see a constantly incrementing value displayed which is the total since the last device reset. If you set DataType to Delta, you will see a number that represents the number of UDPInDatagrams since the last poll.

ShowValue - allows you to specify that a small text box be drawn below any graphical controls that are on the form.

The LED control will show only an LED when the ShowValue property is set to False. If the property is changed to True the LED will be displayed with a small text box below it. The text box will display the last polled value of the SNMP MIB variable.

Value and **StringValue** – are almost exclusively used at run-time to read the last value returned from a device poll. You can use them at design-time to set a value for assisting in design.

If you place a gauge on a form, you can set the initial Value or StringValue so that the needle of the gauge is in a visible range. This value will be overwritten at run-time with the polled value.

VariableType – This will contain the ASN type number of the returned SNMP variable.

Alarm and Threshold Properties

There are several properties that can be set to provide visual feedback of 'normal' and 'alarm' conditions for the variable being polled. The alarm related properties are:

Alarm	MaxThreshold	Alarm Values
Trap	MaxReset	AlarmValues
	MinThreshold	NormalColor
	MinReset	AlarmColor
	NormalValues	

Alarm – tells the control how to evaluate any thresholds that you have set. This property can take on the following values:

0 – Disabled (do not do threshold checking)

1 – Min/Max Thresholds (use MinThreshold/MaxThreshold)

2 – Norm/Alarm Values (Use NormalValues/AlarmValues)

If this property is 0, no thresholds will be checked and the control will default to the color specified by the NormalColor property. If this property is 1 the control will evaluate the polled value against the configured Min/Max thresholds and set its color appropriately. If this property is set to 2, the polled value will be compared against the NormalValues/AlarmValues.

Trap – tells the control whether to send an SNMP Trap packet whenever alarm conditions are set or cleared. If False (default) no SNMP Traps will be sent. If True, SNMP Traps will be sent to OpenView whenever a threshold is crossed that creates an alarm condition, and also whenever a reset value is crossed causing an alarm condition to be cleared.

Threshold Properties for Integer or Counter Variables

One of the methods to indicate alarms is to determine a *normal operating range* for a particular variable. Variables of this type will tend to be integer or counter variables and the value will vary within this defined range. Thresholds can be set on the control so that when the variable value moves outside this normal operating range an alarm condition is created. When the variable value returns into the normal operating range, the alarm condition is cleared. These values will only be checked if the Alarm property is set to '1 - Min/Max Thresholds'.

MaxThreshold – sets a 'high' threshold for the value being polled. When the polled value is *greater than* the value of this property the control will change to the color specified by the AlarmColor property.

MaxReset – sets the value at which an alarm condition should be cleared. If the MaxThreshold property value has been crossed and an alarm condition exists, the polled value must *drop below* the value specified in the MaxReset property before the control will return to the color specified in the NormalColor property.

MinThreshold – sets a 'low' threshold for the value being polled. When the polled value is *less than* the value of this property the control will change to the color specified by the AlarmColor property.

MinReset – sets the value at which an alarm condition should be cleared. If the MinThreshold property value has been crossed and an alarm condition exists, the polled value must *rise above* the value specified in the MinReset property before the control will return to the color specified in the NormalColor property.

Threshold Properties for State Variables

Many variables do not fit into the standard threshold definition. For example, a port on an ethernet hub might have a variable that represents 'link status.' This variable might have several possible 'states'. The variable value might be one of several different values and move from one to another as the system operates. In this case, there could be several 'states' that represent a *normal status* and several that represent an *error status*. These values will only be checked if the Alarm property is set to '2 - Norm/Alarm Values'.

To provide support for these variables, another form of thresholds has been provided with two additional properties. These properties are **NormalValues** and **AlarmValues**.

The **NormalValues** property is used to specify a list of values that represent 'normal' states. When the control is equal to a value in this list, the control will display in the color specified by the NormalColor property.

The **AlarmValues** property is used to specify a list of values that represent 'alarm' states. When the control is equal to a value in this list, the control will display in the color specified by the AlarmColor property.

If the same value is included in both the AlarmValues list and the NormalValues list, an alarm condition will be created. This is a safeguard feature to ensure that an operator will always get alarms if they exist.

The **NormalColor** property is used to set the color of the control when there *is no* alarm condition present.

The **AlarmColor** property is used to set the color of the control when an alarm condition *is* present.

The control will be set to ErrorColor if the variable has a value that is not in the NormalValues or AlarmValues lists.

Table Access Properties

MIB Tables are more complicated to access. An SNMP table is generally accessed by reading rows, or subsets of rows, using the SNMP GET-NEXT command or, if the manager application knows the index of a particular row, the variables are read using the SNMP GET command.

What makes this more difficult is that indexes can be any value or combination of values. For example, in MIB-II the IP Routing Table is indexed by IP Addresses. This means that for you to obtain a row in this table you would have to walk through the whole table, or know the IP Address of the node you are looking for.

In order to make table variables easier to obtain, the following special Table Access Properties are provided.

- InstanceIndex
- TableIndex

InstanceIndex and TableIndex allow you to specify that a control should hold the {n}th entry in a table.

InstanceIndex - contains the index of the variable that was retrieved using the Table Access Properties.

TableIndex - sets the row number of the variable that is to be retrieved.

Each time a control executes its poll cycle it will check:

If TableIndex is zero:

- do regular GET using **MibVariable + InstanceIndex**
- act on returned variable

If TableIndex is non-zero:

- start with MIBVariable
- do a GET-NEXT, **TableIndex** number of times
- put the returned index into **InstanceIndex**
- act on returned variable

This provides you with a method to retrieve any variable in any row, by count, and also to learn the returned index value.

For example, to get the description of the third interface entry in the interfaces table of MIB II, you would set the following properties:

NetAddr:	**192.63.36.1**	(Or the device of your choice)
MibVariable:	**1.3.6.1.2.1.2.2.1.2**	(ifDescr column of the interfaces (table)
PollRate:	**10**	(Every 10 seconds)
Community:	**public**	(Or the Community name for the device)
TableIndex:	**3**	(Third entry)

When the control runs, if this third entry exists, the StringValue property will contain the description, TableIndex will be set to 0, and InstanceIndex will contain the SNMP table index of the third entry.

Advanced Properties

Two additional properties provide advanced tuning of the polling process. These are:

- Retries
- Timeout

Retries – sets the number of SNMP request retries attempted by the control before failing with an error event. The default is -1 which will use the system default. Advanced programmers can use this property to override the system default and tune the polling to their desired rates.

Timeout – sets the amount of time that the control should wait for a response before retrying. The default is -1 which will use the system default. Advanced programmers can use this property to override the system default and tune the polling to their desired rates.

Creating an Application

The custom controls are installed automatically when OpenView is installed. They are placed in the `\windows\system` directory. In order to create an application using the OpenView custom controls, you must do the following:

- Run OpenView.
- Run Visual Basic and add the OpenView custom controls to the Toolbox.
- Add an OpenView custom control to your form.
- Set the appropriate property values for the control.
- Run your program.

If you want menu items in OpenView to run your application, or if you want your application to close when OpenView terminates, you can make entries in XOVSRVR.INI to register your program.

Once you have completed your program, you should compile it so that it will use fewer system resources and be suitable for distribution.

Debugging Visual Basic Applications

When developing applications, it is helpful for testing to execute them from Visual Basic. (Note that running your application from Visual Basic uses considerably more system resources than executing it as an .EXE file. The system may not always behave as expected.)

An easy way to find MIB variable object ID's for use in your application is to run the OpenView SNMP Manager. This application contains the standard MIB II (RFC1213) with textual descriptions of each variable. When you have found the variable you want to use, you can copy the MIB variable object ID to the clipboard and paste it into the MibVariable property of your SNMP control.

Sample Programs

The following sample programs are available in the OV\SAMPLES directory. Make sure that Visual Basic is installed and that OpenView is installed and running.

Sample1 – Get the sysUpTime (a simple MIB variable)

Sample2 – Get a device's MAC address (a MIB table variable)

Sample3 – A simple hub manager

Sample 1. Get the sysUpTime for a device

This example describes how to read and display a MIB variable. You will use the Text control to poll and request a device's "sysUpTime."

1 Start OpenView for Windows.

This will let your application use the OpenView SNMP libraries.

2 Start Visual Basic.

3 Add the HPOVCC.VBX custom control file to your project.

This is the file that contains the custom SNMP controls. It is located in your WINDOWS\SYSTEM directory.

4 Add the Text control to your form.

5 Set the properties on the control.

NetAddr Enter the IP or IPX address of the SNMP device you wish to poll.

MibVariable Enter the object ID of the MIB variable you wish to poll. The sysUpTime is in the system branch of the MIB tree. The object ID is "1.3.6.1.2.1.1.3"

6 Run your application from Visual Basic.

Your application should show a value in the textbox similar to the following:

```
2 days 20 hrs 38 min 31.69 sec
```

Sample 2. Get a MAC address

This example describes how to read and display another MIB variable, "ifPhysAddress". This is a device's MAC address.

Repeat steps **1** through **4** in Sample 1.

1 Set the properties on the control.

NetAddr Enter the IP or IPX address of the SNMP device you wish to poll.

MibVariable Enter the object ID of the MIB variable you wish to poll. The MAC Address is in the interfaces branch of the MIB tree. The object ID is "1.3.6.1.2.1.2.2.1.6."

TableIndex Set this property to 1 for the first row in the interface table.

2 Run your application from Visual Basic.

Your application should show a value in the textbox similar to the following:

```
Hex 08 00 08 49 5A D7
```

Sample 3. Monitor a Hub

This example illustrates a simple application to monitor a hub. Hubview is a compiled program. The source files are also provided.

1 Make XOVSRVR.INI file entries for Hubview.

```
[Programs]
HUBVIEW=c:\ov\samples\sample3\backbox.exe

[CommandIDs]
HPHUBA=&Run
HubView,OV_IDMCONTROL,0x107,OVM_ENABLED,HUBVIEW,,Check the hub
status
```

2 Start OpenView.

The entries in the CommandIDs section of the XOVSRVR.INI file will allow you to run Hubview from the OpenView Control menu.

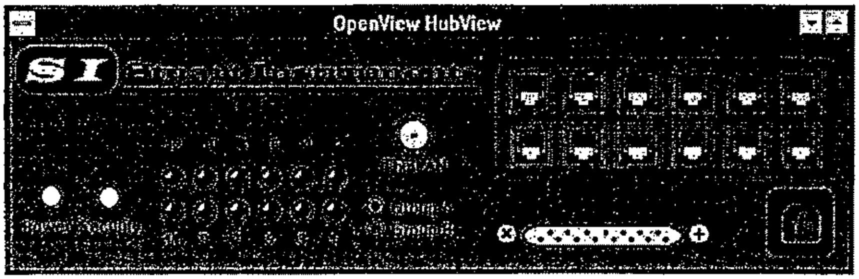
3 Start Visual Basic.

4 Open the Hubview project.

5 Examine the code.

The Hubview program checks the status of 24 ports on the hub. The port ready status is returned as a 24 bit data value. The data is masked to determine the Up/Down status of each port. The port status is then used to select a red or green bitmap to simulate a port indicator. The RJ connectors are used to monitor selected status values for each port.

In order to run this sample, you would need to enter the address and community name of a similar hub installed on your network. You should refer to the MIB files for your network devices to see what status information would be of value to you.



```
Dim s, s2, number, h, M, PortStat(24)
Sub Picture1_Click ()
   hubinfo.Show
End Sub
Sub Port_Click (Index As Integer)
   Traffic.Show
End Sub
Sub Text2_Update (Value As Long)
   s = text2.StringValue
   s2 = text2.StringValue
   h = "0123456789ABCDEF"
s = ((InStr(1, h, Mid$(s, 5, 1)) - 1) * 256 + (InStr(1, h, Mid$(s, 6, 1)) - 1) * 16 + InStr(1, h, Mid$(s,8,
1)) - 1)
 s2 = ((InStr(1, h, Mid$(s2, 9, 1)) - 1) * 256 + (InStr(1, h, Mid$(s2, 11, 1)) - 1) * 16 + InStr(1, h,Mid$(s2,
12, 1)) - 1)
 M = 2048
If option1.Value = True Then
   number = s
Else
   number = s2
 End If
 For I = 1 To 12
  If M > number Then
     portlight(I - 1).Picture = LoadPicture("\ovdev\abdemo\hubview\redlight.ico")
  Else
     portlight(I - 1).Picture = LoadPicture("\ovdev\abdemo\hubview\grnlight.ico")
 number = number - M
  End If
  M = M / 2
 Next I
End Sub
```

Registering Your Visual Basic Application

In order to launch your application from OpenView you must register it with XOVSRVR. This is done with entries in the XOVSRVR.INI file in the \OV directory. Among the sections in this file are Application, Programs, MenuIDs, CommandIDs, and Objects. Not all sections are required for a particular application. You must make an entry in the Programs section. If you want your application to be started by the user you will need an entry in the CommandID section and optionally in the MenuID section.

XOVSRVR.INI File Format

XOVSRVR gets its initialization instructions for its client programs in its initialization file XOVSRVR.INI. XOVSRVR.INI is a text file and follows the WIN.INI file format. These sections are:

- [Application] – contains information about the server, XOVSRVR
- [Programs] – contains information about the client programs
- [MenuIDs] – defines non-terminal menu items. XOVSRVR adds these to the OpenView menus.
- [CommandIDs] – defines terminal menu items. XOVSRVR adds these to the OpenView menus.

The following paragraphs describe the entries in XOVSRVR.INI. Optional entry fields are enclosed in brackets. Entry fields which are enclosed in angle braces and represented in italics are supplied by you. Entry fields are separated by commas. Any optional field which is not present must be delimited by commas. Textual description fields, such as the status string fields, may optionally be enclosed in double quotes ("). However, they **must** be enclosed in double quotes if they contain a comma. Beginning and ending spaces will be stripped from entry fields unless they are textual description fields which are not enclosed in double quotes.

Note that Windows requires that each entry name be unique within its section.

[Applications] Section

The *Description* entry is required in the [Applications] section. It should be set to:

```
[Applications]
Description=XOVSRVR
```

The Description should not be changed.

[Programs] Section

This [Programs] section lists the Visual Basic applications and optionally supplies startup information. The syntax is:

```
TaskName = Program filename[, InitShowCmd]
```

where:

Task Name is the name of your application and is eight characters or less. The Task name is used to reference your application in other parts of the .INI file.

Program filename is the full path and file name of the your application's executable file (ex. OVTEST = c:\ov\ovtest.exe).

InitShowCmd is an optional field to direct XOVSRVR when to start your application. The default value is 0. The values result in the following actions:

0 Do not start the application during XOVSRVR initialization

1 Start during XOVSRVR initialization and display the application window normally

2 Start during XOVSRVR initialization and display the application window minimized

3 Start during XOVSRVR initialization and display the application window maximized

An invalid value of *InitShowCmd* is treated as 0. Programs which are not started during XOVSRVR initialization will be *WinExeced* when one of their menu commands is selected. Their windows will be displayed normally.

Example

```
[Programs]
PROG1=c:\myapp\myexe1.exe
PROG2=c:\myapp\myexe2.exe, 2
PROG3=
```

The entry shows three OpenView applications as clients of XOVSRVR. At least one entry is required for this section. Note that PROG2 will be started when XOVSRVR is loaded, and its window will be initially minimized. PROG3 is a dummy entry. This allows the program to initialize and connect to XOVSRVR but the program will not be launched by XOVSRVR. This is useful during development. When XOVSRVR reads the empty entry, it will add PROG3 to its list of client applications. But it will not start or try to communicate with the application until contact is initiated by the application (when the application is run from Visual Basic).

Menus and Commands

OpenView applications have two types of menu items: those which bring up a popup menu with more menu items, and those which pass a command to the application. The [MenuIDs] section is used for specifying those menu items which bring up a popup or cascading menu. The [CommandIDs] section specifies menu items that result in a command.

MenuIDs Section

The MenuIDs section is used only if you wish your application's menu item to be a part of a non-standard menu or a cascading menu. The syntax is as follows:

```
MenuID=Menu Name,[!]Parent Menu,[Description]
```

where:

MenuID is an identifier to refer to this menu within the .INI file and must be unique within the [MenuIDs] section.

Menu Name is the label to be displayed in the menu. It may be optionally enclosed in double quotes.

[!] is an optional ! character causing a menu separator bar to be drawn above the new menu item.

Parent Menu is the menu under which the new menu will be added.

Description is optional and is the text you will see at the bottom of the OpenView window when the menu item is highlighted. It can be enclosed in double quotes.

Parent Menu is either a menu already described in this section or one of the pre-defined menus listed below.

OV_IDMMAIN	add <*Menu Name*> to the main menu (menu bar)
OV_IDMMONITOR	add <*Menu Name*> to the "Monitor" menu
OV_IDMDIAGNOSE	add <*Menu Name*> to the "Diagnose" menu
OV_IDMCONTROL	add <*Menu Name*> to the "Control" menu
OV_IDMHELP	add <*Menu Name*> to the "Help" menu
OV_IDMFILE	add <*Menu Name*> to the "File" menu
OV_IDMEDIT	add <*Menu Name*> to the "Edit" menu
OV_IDMVIEW	add <*Menu Name*> to the "View" menu
OV_IDMOPTION	add <*Menu Name*> to the "Option" menu
OV_IDMPOPUP	add <*Menu Name*> to the popup menu

<*Parent Menu*> may also be a <MenuID> defined in the <*MenuIDs*> section.

Using OV_IDMMAIN will place your menu on the menu bar and OV_IDMPOPUP will place the menu in the menu that appears when clicking the right mouse button on a symbol. The following example creates a new menu (HubView) on the menu bar and two cascading menus.

Examples

```
[MenuIDs]
HubView=&HubView,OV_IDMMAIN,Hub Manager
Command 1=Command &1,HubView,Start HubView
Command 2=Command &2,HubView,Stop HubView
```

[CommandIDs] Section

This section is used to specify the menu item that a user would use to start your application. A command ID is required so that OpenView can tell XOVSRVR when one of its commands has been selected. XOVSRVR will then tell your application if it is running and has a DDE link established. If your application is not running, XOVSRVR will launch it and tell it of the command as soon as a DDE link with your program is established.

The syntax of a [Commands] entry is:

```
CmdEntry=CommandName,[!]Parent Menu,Command Id,Flags,
Task Name, Object Type[, Description]
```

where:

- *CmdEntry* is an identifier to refer to this command within the .INI file. CmdEntry must be unique within the [CommandIDs] section. You should use an identifier that will not conflict with one from another application.
- *CommandName* is the string which will appear on the menu. It may be optionally enclosed in quotes. A key mnemonic may be added to a menu or command name by preceding the desired letter with an ampersand (&) in the entry.
- *Parent Menu* specifies the menu where *CommandName* will be displayed. This may be a custom menu name from the [MenuIDs] section as well as the standard OpenView menus listed earlier, other than the main menu.
- *Command ID* is a number assigned by you that is unique to your application that refers to the menu item. It can be decimal, hex, or octal. It must be greater than 100. (Values less than 100 are reserved for OpenView.) This value is used by your application. When the user selects *CommandName*, *Command ID* will be passed to your application as a parameter in the XOV custom control's command event procedure.
- *Flags* is a required field. *Flags* identifiers are:

OVM_ENABLED	command initially enabled
OVM_DISABLED	command initially disabled (default)
OVM_CHECKED	command initially checked
OVM_UNCHECKED	command initially not checked (default)
OVM_OBJSPEC	command is object specific
OVM_NOT_OBJSPEC	command is not object specific (default)

Flag identifiers which are not mutually exclusive may be or'ed together using a vertical bar:

OVM_DISABLED | OVM_OBJSPEC

- *Task Name* specifies the application defined in the [Programs] section that has registered for this command.
- *Object Type* is the object type that will enable (ungrey) the menu item if object specific.
- *Description* is optional and contains text to be displayed on the OpenView status bar.

A place holder (two commas ",,") is required if a missing field is not the last field.

Example

```
[MenuIDs]
SMGR=&SNMP Manager,OV_IDMCONTROL,Mib manager tool

[CommandIDs]
SMGR1=&Define Query...,SMGR,0x104,OVM_ENABLED,MIB,,Get or Set Mib variables
SMGR2=&Select Query...,!SMGR,0x101,OVM_ENABLED,MIB
SMGR3=&Manage Database...,SMGR,0x100,OVM_ENABLED,MIBCOMP
MyHelp=MyHelp...,OV_IDMHELP,0x102,OVM_ENABLED|OVM_OBJSPEC,MYTASK
```

Note the use of key mnemonics in the first four entries. A separator bar will be inserted before the Select Query entry. The &Define Query entry required a place holder (,,) for the missing fields.

The entry to add cascading menus and an additional command for the HubView example (see MenuIDs section) would be:

```
[CommandIDs]
HubView0=Command &3,HubView,0x700,OVM_ENABLED,,,HubView Utilities
HubView1=&Get Data,Command 1,0x701,OVM_ENABLED,,,Fresh Start,SYM_HUB
HubView2=&Report,Command 1,0x702,OVM_ENABLED,,,Print Report HubView
HubView3=&Sleep,Command 2,0x703,OVM_ENABLED|OVM_CHECKED,,,Pause HubView
HubView4=&Quit,Command 2,0x704,OVM_ENABLED,,,End HubView
```

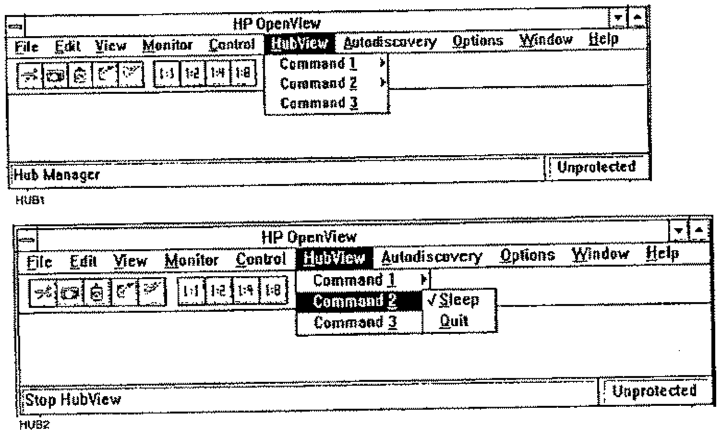


Figure 7-1 Examples of Menu and Command items

[Objects] Section

The Objects section registers objects to applications running under the server. The format for entries in this section is:

```
Object Name=[Application Name][,Application Name]...
```

Object Name is either one of the pre-defined names for the standard objects (listed below) or the number of an object type as listed in OVWIN.INI for user defined symbols.

Application Name is the name of the application (defined in the Programs section) to which the object is to be registered. You may specify the object be registered to multiple applications by separating each name with a comma.

The pre-defined symbol names are:

SYM_SUBPICTURE
SYM_FSERVERPICT
SYM_MEDCOMPPICT
SYM_LARGE_COMPUTER
SYM_MEDIUM_COMPUTER_2
SYM_GOTOSUBPICT
SYM_PCPICT
SYM_MEDIUM_COMPUTER
SYM_SMALL_COMPUTER
SYM_DBSERVER
SYM_LINE
SYM_LINE_1
SYM_LINE_2
SYM_LINE_3
SYM_LINE_4

SYM_PC
SYM_PSERVER
SYM_DOCSERVER
SYM_ANALYZER
SYM_ROUTER
SYM_PBX
SYM_BRIDGE
SYM_HP_DTC
SYM_LOCAL_PRINTER
SYM_FSERVER
SYM_T1_MUX
SYM_MODEM
SYM_GATEWAY
SYM_DTC
SYM_X_25
SYM_NET_PRINTER
SYM_SNA
SYM_HUB
SYM_LINE_5
SYM_LINE_6
SYM_LINE_7
SYM_LINE_8
SYM_LINE_9
SYM_LINE_10
SYM_LINE_11
SYM_LINE_12

The following example will register the hub symbol to the Hubview application:

```
[Objects]
SYM_HUB=HUBVIEW
```

This completes the registration process.

Adding Features To Your Application

In order for you to make use of the additional program launch and terminate functions you must register your application in the XOVSRVR.INI file. To terminate your program from OpenView, you must add the XOV.VBX control to your form.

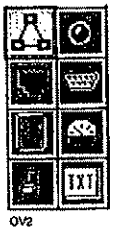
When you run your program under Visual Basic, if your XOV custom control Enabled property is TRUE, the control will initiate communication with XOVSRVR and your program will run.

You can make your application shut down upon receiving an End Session message for OpenView by adding the XOV.VBX control to your application and placing an End statement in the EndSession event of the OpenView custom control.

1 Add the needed entries to XOVSRVR.INI to add your application's menu items and commands to OpenView's menu.
2 Run OpenView to start XOVSRVR.
3 Run Visual Basic and select File, Add File, \windows\system\XOV.VBX.

4 Repeat for the SNMP controls, HPOVCC.VBX.

The controls will be added to the tool box.



The XOV custom control is similar to the Visual Basic Timer control in that it is not sizable and is invisible at run-time. The main function of this control is to allow OpenView to send a message to your application when OpenView is terminated.

5 Click OK.

The XOV control will be added to the Visual Basic tool box.

6 Select the XOV control and drag it to your form.

Since this control is visible only at design time, it doesn't matter where you place it on the form. The main form is a good choice to contain XOV. One of the properties of the XOV control is the *enabled* property. This property is automatically set to TRUE at runtime. Note that this communication will only happen if OpenView is running. Only one instance of this control can be added to your program.

7 Select the Properties command in the Visual Basic Windows menu.

This will display the Properties window for the XOV control.

You must match the TaskName property with the name of your application as defined in the [Programs] section of XOVSRVR.INI (OVTEST for example).

8 Enter the name of your application (OVTEST) in the TaskName field.

9 Add an END statement to the End_Session procedure of the XOV control.

A

OpenView Command and Tool Reference

This appendix lists OpenView menus commands and their functions, and lists the tools in the Toolbar and the tool functions.

Menu Commands

The following tables list the OpenView commands by menu.

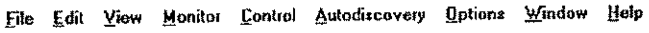


Figure A-1 OpenView menus

File Menu Commands

Command	Description
New	Start a new (empty) map.
Open...	Open an existing map file.
Save	Save the map using the current file name.
Save As...	Save the map to a different file name.
Check Map...	Check the map for errors.
Print Submap	Print the current submap portion displayed on screen.
Print Object List	Print a list of all objects in the map, sorted by submap.
Print Setup...	Configure printing.
Load Selection List...	Select a set of objects whose names have been saved in a file.
Save Selection List As...	Save a set of object names to a file.
Exit	Exit OpenView.

Edit Menu Command

Command	Description
Add	Display the Add toolbox.
Cut	Copy the last selected object to the clipboard and delete from submap.
Copy	Copy the last selected object to the clipboard.
Paste	Copy clipboard contents to the next selected position on a submap.
Delete	Delete a selected object from the map.
Describe...	Add a name, label, addresses, and notes to the selected object.
New Submap...	Create a new submap.
Rename Submap...	Rename the current submap.
Clear Submap	Erase all objects in the current submap.
Delete Submap...	Delete a submap and its contents from the map.
Set Background Image...	Select a background file for the current submap.
Clear Background Image	Delete the reference to a background file for the current submap.
Set Home Submap...	Define the top level ("Home") submap for the map.

View Menu Commands

Command	Description
Zoom 1	Set the display to normal.
Zoom 1/2	Set the display to 1/2 scale.
Zoom 1/4	Set the display to 1/4 scale.
Zoom 1/8	Set the display to 1/8 scale.
Pan...	Shift the view in the current submap.
Set Grid...	Select a grid spacing as a drawing aide.
Show Grid	Display the drawing grid on the current submap.

Monitor Menu Commands

Command	Description
Describe...	Add a name, label, address(es), and notes to the selected object.
Status Legend...	Display the alarm status colors.
Alarm Log...	Display the Alarm Log.
Ping	Ping selected device.
Customize Traps...	Change default trap settings.
HP TopTools	Run TopTools application.
Polling	Poll network devices.
View Polling List...	List devices in current polling list.
Add Device(s)	Add devices to poll to current list.
Remove Device(s)	Remove devices to poll from currents list.
Configure System Defaults...	Change interval and alarm action defaults for a device.
Configure Device Parameters...	Change interval and alarm action defaults for a device.
Start Polling	Initiate polling of network devices.

Control Menu Commands

Command	Description
Browse Web Home Page	Display Web Home Page associated with device.
Browse Web Management Page	Display Web Management Page associated with device.
SNMP Manager	Run the SNMP Manager application.
Define Query...	Sets query variables and display options.
Select Query...	Selects and performs a query.
Manage Database...	Selects MIB files to add to the database.
DMI Manager	Run the DMI Manager Application.

Autodiscovery Menu Commands

Command	Description
Configure	Configure the Autodiscovery facility.
Discovery Schedule...	Schedule the discovery process.
Discovery Networks...	Select networks to be searched.
User-Specified Devices...	Specify additional devices.
Ignored-Devices...	Specify devices to ignore in layout.
Device Classes...	Specify classes of devices to be ignored in layout.
Discover	Discover network devices.
Discovery Manager...	Display the Discovery Manager Dialog.
Discover Routers	Initiate router discovery.
Update Host Names	Change the Host names.
Clear Discovery Database	Delete the current discovery database.
Dump Discovery Database	Copy the discovery database.
Discover Topology	Discover network segments
Layout	Layout discovered devices.
Do Basic Layout	Begin layout.
Basic Layout Options...	Configure layout options.
Configure Preferred Name...	Enter names for nodes.

Options Menu Commands

Command	Description
Protect Map...	Enable or disable manual editing of a map file using a password.
Log In...	Log In and select a security level.
Log Out	Disables all menu items in OpenView and locks the map.
Set Password...	Sets or changes passwords.
Customize HP OpenView...	Select maximum symbol and message counts for a map, specify object description operation, log file, and default map.
Customize Alarms...	Select alarm frequency, sort order, sounds, and display propagation.
Customize Device Access...	Associate control information with a network address.

Window Menu Commands

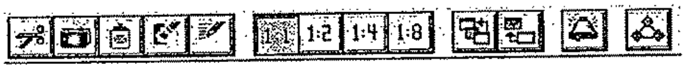
Command	Description
Home Submap	Display the home submap.
Previous Submap	If you opened a submap symbol to display the current submap, this command will return you to the previous submap.
Locate Submap...	List all submaps in the map file and display the one selected.
Locate Object..	List all objects in the map file and display the submap containing the one selected.
Extend Locate...	Search for map objects using a variety of selection criteria.

Help Menu Commands

Command	Description
HP OpenView Index...	Access on-line help for OpenView.
About...	Display copyright and version information for OpenView.

Toolbar

OpenView displays a toolbar at the top of the main window. The toolbar provides quick access to frequently used functions. The toolbar buttons are described in the following table.



pg158.tif

Figure A-2 Toolbar Buttons

Toolbar Functions

Tool	Description
Scissors	Cut (same as in the edit menu).
Camera	Copy "
Paste	Paste "
Eraser	Delete "
Pencil	Describe "
1:1	Zoom 1 (same as in the View menu).
1:2	Zoom 1/2 "
1:4.	Zoom 1/4 "
1:8	Zoom 1/8 "
Home Submap	Display the Home submap (same as in the Window menu).
Previous Submap	Display the previous submap (same as in the Window menu).
Alarm Bell	Display the alarm log. Icon color reflects unacknowledged alarms.

B

CONFIGURING NOTIFY! CONNECT

OpenView is shipped with a remote paging application Notify! Connect from Ex Machina Inc. The application can receive messages from OpenView's alarm system and use a modem to generate a page at a remote location. When OpenView is installed the Notify! Connect application is automatically installed. OpenView makes entries in the OVWIN.INI file and sets several of the Notify! Connect parameters to default values during OpenView's installation. Before you can send pages you must update these values to match your paging configuration.

Values that must be entered or changed are:

- user to be paged (default is OPENVIEW)
- paging service (default is SKYPAGE)
- Pager ID number of user (default is 0)
- format of paging message (see OVWIN.INI)

OVWIN.INI Configuration

In order to have an alarm generate a page you must make an entry in the OVWIN.INI file under the [OVAlarm] section. The entry instructs OpenView to send a service request to a program. OpenView makes the following default entries in the OVWIN.INI file:

```
[OVAlarm]
;RunCritical=>C:\OV\notify\connect.exe,NOTIFY,SendPage;OpenView,Critical Alarm: %o - %a
;RunMajor=>C:\OV\notify\connect.exe,NOTIFY,SendPage;OpenView,Major Alarm: %o - %a
;RunInformational=>C:\OV\notify\connect.exe,NOTIFY,SendPage;OpenView,Info Alarm: %o - %a
```

The entries are disabled until the semicolons (;) are removed. When the semicolons are removed and OpenView is restarted, OpenView will request a page be sent to a user named "OpenView" if a major, critical, or informational alarm occurs. (OpenView generates an informational alarm when a users logs in.) Refer to chapter 4 of this manual for more information.

Notify! Configuration

OpenView makes the following default entries in the Notify database:

- user to be paged: OPENVIEW
- type of paging service: SKYTEL
- modem settings

To configure Notify! Connect for use you must enter the correct settings for your paging system.

1 Run Notify! Connect by double-clicking its icon in the HP OpenView program group.

Select the Services function and add your service if necessary and correct any of the settings as needed.

2 Select the Address function and edit the entries for the user "OpenView" (Service, Pager type, Pager ID etc.).

3 Select the Server function and make any required changes to the communications settings.

It is recommended that a business dialing prefix be set here if required.

4 Use the Test function to verify that your pager system is working before using the pager through OpenView.

5 After you have tested the paging configuration in Notify! you can test it with an informational alarm in OpenView.

Logging into OpenView can be used to trigger an informational alarm.

You may wish to have Notify! Connect running as a background server whenever you are running Windows. You can do this by copying the Notify! icon from the HP OpenView group into the Startup group. If you select the properties for this icon you can select "Minimize on Startup" to have Notify! display as an icon when started. This will prevent the main Notify! menu from displaying when an alarm generates a page.

Additional information for configuring and using Notify! is contained in the Notify! Connect documentation.

INDEX